Unique Pointer Solutions

Copyright © James Raynard 2021

std::unique_ptr

- Briefly describe how std::unique_ptr is implemented
  - unique_ptr is a class which has a traditional pointer as a private data member
  - It has public member functions which implement some of the features of traditional pointers
- In terms of memory usage and efficiency, how does unique_ptr compare to traditional pointers?
  - There is no extra overhead from using a unique_ptr instead of a traditional pointer

Copyright © James Raynard 2021

std::unique_ptr and RAII

- Explain how unique_ptr follows the principles of RAII
  - The allocated memory is managed by the class
  - It is acquired or allocated in the class's constructor
  - It is released in the class's destructor
  - The class manages transfer of ownership of the traditional pointer from one object to another

Copyright © James Raynard 2021

unique_ptr initialization

- Write a simple program that creates and initializes a unique_ptr object and performs some operations on it
- What changes would you need to make your program compile under C++11?
- (Optional) Put your compiler into C++11 mode and check your answer to the previous question

Copyright © James Raynard 2021

Returning unique_ptr

- What happens when a unique_ptr is returned from a function?
  - The function creates a unique_ptr object as a local variable
  - This will internally allocate memory for its pointer member
  - When the function returns, the unique_ptr will be moved into the return space and then moved into a variable in the caller
  - When the caller's variable is destroyed, the memory allocated will be released automatically
  - At each stage, unique_ptr's move constructor will be called to transfer the allocated member from one object to another

Copyright © James Raynard 2021

Returning unique_ptr

- What are the advantages of returning unique_ptr instead of a traditional pointer?
  - The allocated memory is owned by the caller's unique_ptr object and can only be accessed through the unique_ptr interface
  - The memory will automatically be released when the caller's object goes out of scope (unless it is moved again)
  - No doubts about who is responsible for releasing the memory

Copyright © James Raynard 2021

Returning unique_ptr instance

- Write a function which creates a unique_ptr local variable which is returned by the function
- Write a program that calls the function and prints out the data in the returned object

Copyright © James Raynard 2021